Sinkronisasi *Content E-learning* Terdistribusi Berbasis Model Komunikasi *Indirect* Menggunakan Sistem *Publish-Subscribe*

Sufrendo Saputra, Royyana Muslim Ijtihadie, dan Waskitho Wibisono
Jurusan Teknik Informatika, Fakultas Teknologi dan Informasi, Institut Teknologi Sepuluh Nopember (ITS)
Jl. Arief Rahman Hakim, Surabaya 60111 Indonesia
e-mail: roy@if.its.ac.id

***Abstrak*—Sinkronisasi *content* antar *e-learning* memungkinkan beberapa *e-learning* memiliki *content* yang sama secara konsisten. Perubahan *content* pada salah satu *e-learning* akan membuat sistem memastikan *e-learning* lain mengetahui perubahan tersebut. Model komunikasi yang memungkinkan adanya sinkronisasi ini merupakan komunikasi *indirect* berbasis *publish-subscribe*. Setiap *e-learning* memiliki *content*-nya masing-masing yang secara otomatis akan di-*publish* oleh sistem. *E-learning* lain yang tergabung dalam sistem sinkronisasi kemudian dapat memilih *content* mana yang ingin di-*subscribe*. Jika terdapat perubahan pada sebuah *content*, dan *content* tersebut memiliki *subscriber*, maka sistem akan memberitahu *subscriber* bahwa telah terjadi perubahan pada *content*. Teknologi utama yang digunakan dalam sistem ini adalah Moodle, PHP, dan Java. Moodle sebagai modul yang digunakan untuk mensimulasikan *e-learning*. PHP dan Java sebagai *framework* dari sistem sinkronisasi. Model komunikasi yang digunakan merupakan komunikasi *indirect* berbasis *publish-subscribe*. Model komunikasi ini menempatkan sebuah perantara bagi komunikasi antar *e-learning*.**

***Kata kunci*—*e-learning*, java, *publish-subscribe*, sinkronisasi**

I. PENDAHULUAN

SAAT ini kita hidup di zaman di mana sistem terdistribusi sudah menjadi bagian dari kehidupan kita sehari-hari. Sistem terdistribusi sendiri didefinisikan sebagai sebuah sistem di mana komponen-komponen yang berlokasi pada sebuah jaringan komputer saling berkomunikasi dan mengoordinasikan aksi mereka dengan cara bertukar pesan. Salah satu contoh aplikasi modern dari sistem terdistribusi dalam bidang pendidikan adalah *e-learning* atau sistem pembelajaran elektronik.

George Colouris dalam "*Distributed Systems: Concepts and Design*" menyebutkan bahwa dalam sistem terdistribusi terdapat tiga tipe paradigma komunikasi antar entitas [1]. Salah satunya adalah *indirect communication* atau komunikasi tak langsung. Sifat khusus dari *indirect communication* adalah penggunaan perantara dalam pengiriman pesan. Penggunaan perantara ini melahirkan dua karakteristik penting yaitu *space uncoupling* dan *time uncoupling*. Patrick Th. Eugster dalam "*The Many Faces of Publish/Subscibe*" bahkan menyebutkan bahwa sistem *publish-subscribe,* salah satu model dari paradigma *indirect communication,* memiliki satu karakteristik tambahan yaitu *syncronization uncoupling* [2].

Sifat-sifat yang dimiliki oleh paradigma *indirect communication* dapat dimanfaatkan untuk sinkronisasi dalam sistem terdistribusi. Kasus sinkronisasi yang diangkat dalam Tugas Akhir ini adalah sinkronisasi *content* antar *e-learning*.

Salah satu alasan mengapa sinkronisasi *content* antar *e-learning* dirasa perlu adalah untuk menyeragamkan kualitas *content*. Materi pembelajaran yang disusun oleh para ahli yang sudah terpercaya di bidangnya tentu akan dirasa lebih berkualitas. Dengan adanya sikronisasi, bahan ajar tersebut nantinya dapat digunakan pada platform *e-learning* milik instansi lain sebagai upaya untuk meningkatkan kualitas pembelajaran. Tenaga pengajar tidak perlu dipusingkan dengan penyusunan materi dan dapat fokus pada yang sudah ada.

Sinkronisasi dilakukan menggunakan sistem *publish-subscribe,* sebagai salah satu model dari paradigma *indirect communication,* karena beberapa sifatnya yang telah disebutkan di atas. Alasan lainnya karena model *direct communication* dirasa bukan menjadi prioritas dalam kasus sinkronisasi *content*. Alur pengiriman data dalam proses sinkronisasi tidak perlu dibebankan kepada perangkat *e-learning* melainkan cukup difokuskan pada satu pihak yang berfungsi sebagai *event router*.

Sebagai batasan, platform *e-learning* yang digunakan pada Tugas Akhir ini adalah Moodle, dan sinkronisasi yang dimaksudkan meliputi *activity*/*resource* pada Moodle yaitu, *page* dan *file*.

`Makalah ini akan membahas mengenai perancangan sistem sinkronisasi *content e-learning* terdistribusi berbasis model komunikasi *indirect* menggunakan sistem *publish-subscribe*.

II. TINJAUAN PUSTAKA

A. *Indirect Communication*

Indirect communication didefinisikan sebagai komunikasi antar entitas dalam sebuah sistem terdistribusi melalui sebuah perantara tanpa ada hubungan langsung antara pengirim dan penerima. Sifat spesifik perantara berbeda-beda dari satu pendekatan ke pendekatan lainnya. Sebagai tambahan, sifat spesifik pengirim dan penerima antar sistem juga dapat memiliki perbedaan yang signifikan[1].

Penggunaan perantara dalam paradigma *indirect* communication melahirkan tiga karakteristik penting yaitu *space uncoupling*, *time uncoupling*, dan *synchronization uncoupling*.

B. Sistem Publish-Subscribe

Sistem *publish-subscribe* merupakan sistem di mana *publisher* memublikasikan *event* terstruktur kepada *event service* dan *subscriber* menyatakan ketertarikan terhadap *event* tertentu melalui *subscription* yang dapat berupa pola sembarang dari *event* terstruktur tersebut. Tugas dari sistem *publish-subscribe* adalah untuk mencocokkan *subscription* dengan *event* yang dipublikasikan dan menjamin pengiriman notifikasi *event* yang benar. Sebuah *event* bisa saja dikirimkan ke banyak *subscriber*, dan karena itulah pada dasarnya *publish-subscribe* merupakan sebuah paradigma komunikasi *one-to-many*.

C. E-learning

E-learning mengacu pada penggunaan internet atau teknologi nirkabel untuk menyediakan berbagai macam solusi pelatihan. Dalam lingkungan *e-learning*, murid berinteraksi dengan bahan ajar, instruktur (guru dan/atau dosen) dan murid (mahasiswa) lain dari berbagai lokasi dan seringkali pada waktu yang berbeda-beda menggunakan teknologi jaringan komputer. Pada dasarnya, *e-learning* menawarkan fleksibilitas yang tinggi terhadap kapan dan bagaimana proses pembelajaran berlangsung.

D. Moodle

Moodle merupakan *Course Management System* (CMS) yang dikembangkan oleh Martin Dougiamas. Nama Moodle merupakan akronim dari *Modular Object-Oriented Dynamic Learning Environment*. Beberapa kelebihan yang dimiliki Moodle dibandingkan dengan CMS lainnya antara lain sifatnya yang *free* dan *open source*, desain yang berfilosofi pada pendidikan, serta memiliki komunitas yang besar dan aktif dalam mengembangkan perbaikan dan fitur-fitur baru [3]. Sebuah *course* dalam Moodle dapat terdiri atas bermacam-macam komponen atau yang biasa disebut dengan *activity/resource*.

Activity/resource merupakan nama umum untuk sekumpulan fitur dalam sebuah *course* Moodle. Anda dapat menbayangkan *activity* sebagai kumpulan modul yang melengkapi sebuah *course*. Biasanya *activity* merupakan sesuatu yang akan dilakukan murid dimana mereka berinteraksi dengan murid lain atau dengan sang instruktur. Ada berbagai macam tipe *activity* dalam instalasi standar Moodle yang dapat dilihat pada menu *add an activity*. Dua jenis *activity* yang akan coba disinkronisasikan dalam Tugas Akhir ini adalah *page* dan *file*.

E. Java

Java adalah bahasa pemrograman berorientasi objek yang dikembangkan oleh Sun Microsystems sejak tahun 1991. Bahasa ini dikembangkan dengan model yang mirip dengan bahasa C++ dan Smalltalk, namun dirancang agar lebih mudah dipakai dan platform independent, yaitu dapat dijalankan di berbagai jenis sistem operasi dan arsitektur komputer.

Kompiler dan interpreter untuk program Java berbentuk JDK (*Java Development Kit*) yang diproduksi oleh Sun Microsystems. Interpreter untuk program Java sering juga disebut Java Runtime atau JVM (*Java Virtual Machine*). Interpreter Java tanpa kompilernya disebut JRE (*Java Runtime Environment*). Untuk mengembangkan program Java dibutuhkan JDK, sementara untuk menjalankan program atau aplikasi berbasis Java cukup dengan JRE saja.

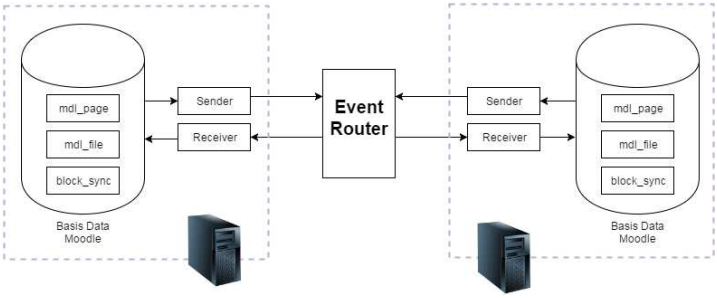
F. Event router Avis

Avis merupakan sebuah *event router*. Avis menyediakan layanan *event routing* berbasis *publish-subscribe* yang menerapkan implementasi Elvin hasil karya Mantara Software. Beberapa kelebihan *event router* antara lain pengiriman pesan dengan format yang fleksibel dan dapat dijalankan pada platform manapun yang mendukung Java 5 ke atas.

III. DESAIN DAN IMPLEMENTASI

A. Arsitektur Sistem

Sistem terdiri atas beberapa mesin yang menjalankan Moodle. Mesin ini akan saling terhubung dengan sebuah *server event router*. Pada tiap mesin akan terdapat 2 buah program Java yang bernama *Sender*.java dan *Receiver*.java. Kedua program ini yang akan berkomunikasi dengan *event router* untuk menjalankan proses sinkronisasi. Gambaran besar arsitektur sistem dapat dilihat pada Gambar 1.



Gambar 1. Arsitektur sistem sinkronisasi.

Program *sender* secara berkala akan melakukan pengecekan pada basis data Moodle untuk menetapkan apakah telah terjadi pembaruan. Untuk setiap pembaruan yang terdeteksi, *Sender* akan mengirimkannya kepada *event router*. *Event router* kemudian akan meneruskan data dari *sender* kepada *receiver* milik entitas yang menjadi *subscriber* untuk *content* tersebut. Begitu diterima, *receiver* akan melakukan pembaruan pada basis data Moodle. Tiap entitas dapat menjadi *publisher* sekaligus *subscriber*.

Komunikasi antar entitas berlangsung secara tidak langsung (*indirect communication*). Dalam arti *sender* milik sebuah entitas tidak pernah berkomunikasi secara langsung dengan *receiver* milik entitas lain. Komunikasi ini ditangani oleh *event router* Avis.

Model *subscription* yang digunakan pada sistem ini adalah *content-based subscription*. Di sisi *publisher*, semua pesan dari *Sender* akan diteruskan ke *event router*. *Sender* tidak mengetahui siapa saja yang akan mendapat pesan tersebut. Sedangkan di sisi *subscriber*, *Receiver* terus menerus mendengarkan pesan dari *event router*.

B. *Proses Sinkronisasi*

Proses ini dapat dibagi menjadi beberapa tahap sebagai berikut: proses *subscribe*, menentukan pembaruan pada *content* milik *publisher*, *publisher* mengirimkan pembaruan kepada *event router*, *event router* menotifikasi *subscriber* mengenai adanya pembaruan sekaligus mengirimkan isi pembaruan, *subscriber* menerapkan pembaruan.

Subscriber pertama kali akan memilih *content* apa yang akan di-*subscribe*. Informasi *subscription* akan disimpan dalam tabel block_sync. *Receiver* akan menggunakan informasi dalam tabel tersebut untuk menghasilkan *subscription expression*. Dari sana *event router* akan menotifikasi *receiver* jika terdapat pesan yang cocok dengan *subscription expression* yang dimiliki *receiver*. Informasi *subsciption* yang digunakan *Receiver* untuk menghasilkan *subscription expression* berupa *indeks content* dan status *subscription*.

Indeks *content* merupakan id unik yang diberikan kepada tiap *content* Moodle yang tergabung dalam sistem. Setiap entitas Moodle yang tergabung dalam sistem mengenali id ini. Sedangkan status *subscription* menentukan *content* mana yang di-*subscribe* oleh Moodle. Saat terjadi pembaruan pada *content* milik *publish*er, *sender* akan mengirim pesan kepada *event router*. Pesan tersebut berupa informasi *content* yang mengalami pembaruan beserta dengan isi pembaruan. Informasi *content* yang dikirim berupa indeks *content* yang dapat dikenali oleh *receiver*.

Event router kemudian menotifikasi *receiver*. Jika *event* yang terjadi (pembaruan pada *content* yang di-*subscribe*) sesuai dengan *subscription expression* yang ada pada *Receiver*, maka *receiver* akan menerima dan menerapkan pembaruan ke dalam basis data Moodle.

C. *Metadata*

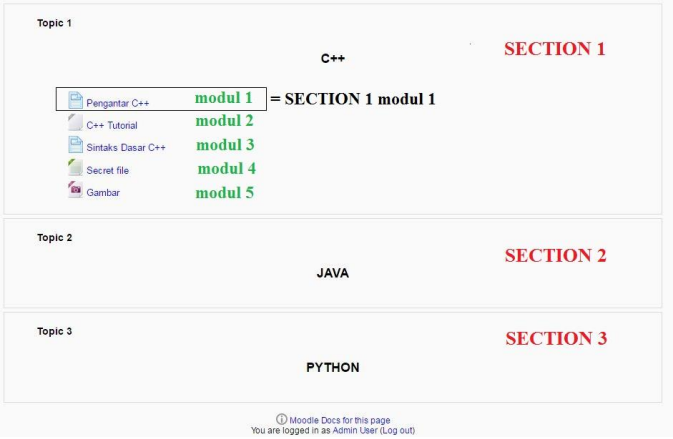
Metadata meliputi data pendukung yang dikirimkan bersama dengan notifikasi kepada *event router*. Data ini bertujuan agar nantinya penerima notifikasi dapat mengenali notifikasi yang didapat dan menerapkan data pembaruan dalam basis datanya secara akurat. Salah satu metadata yang digunakan adalah data dari tabel block_sync untuk mengenali tiap *course* yang ada pada sistem sinkronisasi. Struktur dari tabel block_sync dapat dilihat pada Tabel 1.

Tabel 1.
Struktur tabel block_sync

Data	Penjelasan
tid	Merujuk pada id *course* pada tabel mdl_*course*
sid	Merupakan id unik yang diberikan untuk semua *course* Moodle yang tergabung dalam sistem
owner	Pemilik atau pembuat *course*
coursename	Nama *course*
status	Status *subscribtion* (0=pemilik asli *course*;1=bukan pemilik asli *course*, tidak men-*subscribe*; 2=bukan pemilik asli *course*, men-*subscribe*, 3=pemilik asli *course*, *course* ini baru saja dibuat/belum tersinkronisasi dengan tabel mdl_block_sync lain)

Selain data dari tabel di atas, *publish*er juga mengirimkan metadata yang berisi informasi *content* apa yang diperbarui. Data ini dikirimkan dalam dua nilai yang bernama *section* dan modul. Cara mendapatkan kedua nilai ini ditunjukkan melalui Gambar 2.

Pada Gambar 2, misalkan terjadi pembaruan pada *content* bernama "Pengantar C++", maka *publisher* akan memberikan notifkasi bahwa telah terjadi perubahan untuk *content* dengan indeks *section*=1 dan modul=1.



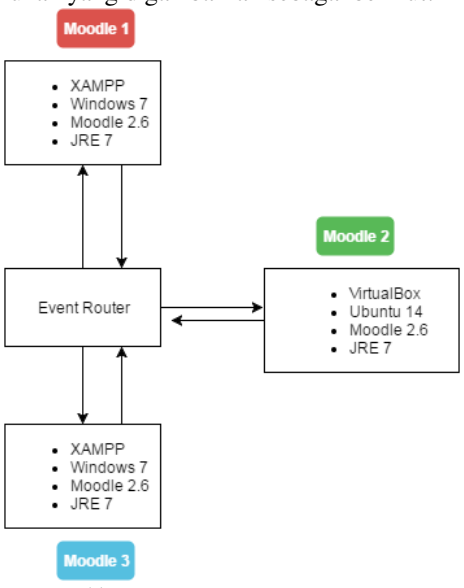
Gambar 2. Menentukan nilai *section* dan modul.

IV. UJI COBA DAN EVALUASI

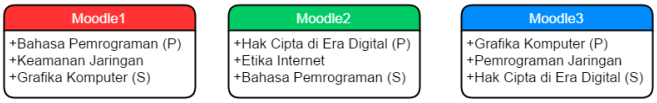
Pada bab ini akan dijelaskan uji coba yang dilakukan pada aplikasi yang telah dikerjakan serta analisa dari uji coba yang telah dilakukan. Pembahasan pengujian meliputi lingkungan uji coba, skenario uji coba yang meliputi uji kebenaran dan uji kinerja serta analisa setiap pengujian.

A. *Lingkungan Uji Coba*

Lingkungan uji coba menjelaskan lingkungan yang digunakan untuk menguji implementasi pembuatan sistem. Lingkungan uji coba meliputi perangkat keras dan perangkat lunak yang digambarkan sebagai berikut.



(a)



(b)

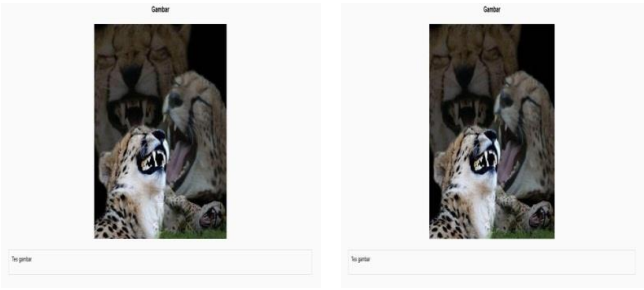
Gambar 3. Arsitektur uji coba (a) dan struktur *course* yang dimiliki tiap Moodle (b).

Pada arsitektur di atas terdapat dua Moodle yang dijalankan melalui *virtual host* XAMPP dan satu Moodle melalui *virtual machine* VirtualBox. Semua mesin menggunakan instalasi Moodle versi 2.6 dan *Java Runtime Environment* (JRE) 7.

B. Skenario 1

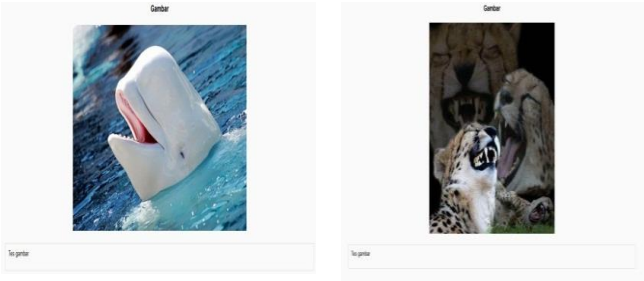
Uji coba ini dilakukan untuk menguji apakah fungsionalitas sistem telah diimplementasikan dengan benar dan berjalan sebagaimana mestinya. Uji coba akan didasarkan pada salah satu skenario untuk menguji kesesuaian dan kinerja sistem sinkronisasi.

Pengujian fungsionalitas dilakukan antara Moodle1 dan Moodle2. Moodle2 men-*subscribe course* Bahasa Pemrograman milik Moodle1. Pada Moodle1 akan dilakukann perubahan pada salah satu *file* yang bernama gambar1.jpg. Gambar ini akan diganti dengan gambar baru dan ketika proses sinkronisasi berjalan, maka *file* gambar1,jpg pada Moodle2 akan ikut berubah.



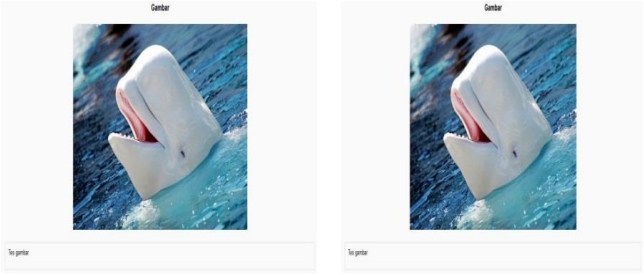
(a) (b)

Gambar 4. Perbandingan gambar1.jpg pada Moodle1 (a) dan Moodle2 (b) sebelum dilakukan perubahan



(a) (b)

Gambar 5. Perbandingan gambar1.jpg pada Moodle1 (a) dan Moodle2 (b) ketika dilakukan perubahan pada Moodle1



(a) (b)

Gambar 6. Perbandingan gambar1.jpg pada Moodle1 (a) dan Moodle2 (b) ketika proses sinkronisasi telah dilakukan.

C. Skenario 2

Pada skenario 2 akan dilakukan uji performa untuk mengetahui besaran data dan *delay* dari proses pengiriman data dari *publisher* hingga sampai ke *subscriber*.

Tabel 5.
Uji performa pengiriman data.

***Content* Yang Mengalami Pembaruai**	**Tipe Data**	**Ukuran Data (*byte*)**	**Rata-Rata Delay (*byte/ms*)**
Pembaruan "Pengantar C++"	teks	1716	602.57
Pembaruan "Helloc++.cc"	teks	214000	485.21
Gambar1	gambar	581000	441.85
Gambar2	gambar	1090000	453.71

V. KESIMPULAN

Dari hasil uji coba yang telah dilakukan terhadap fungsi-fungsi dan kinerja sistem, kesimpulannya adalah sebagai berikut:

1. Sinkronisasi *content* antar LMS Moodle mampu dijembatani oleh metode komunikasi *publish subscribe* dengan menggunakan *event routing*.
2. Dari hasil pengujian pada bab uji coba dan evaluasi, didapat kecepatan dan ukuran data yang dikirimkan dari beberapa kondisi yang berbeda yang telah ditentukan sebelumnya.

DAFTAR PUSTAKA

[1] G. Coulouris et al., *Distributed Systems: Concept and Design*, 5th ed. Boston, MA: Addison-Wesley, 1998.

[2] P. Th. Eugster et al. (2003, June). "The Many Faces of *Publish Subscribe*." ACM Computing Surveys. [Online]. 35(2), pp. 114-131. Available: http://dl.acm.org/citation.cfm?id=857078 [Feb. 28, 2014].

[3] J. Cole and H. Foster, *Using Moodle: Teaching with the Popular Open Source Management System*, 2nd ed. California: O'Reilly, 2007.